WebDSL

A Domain-Specific Language for Dynamic Web Applications

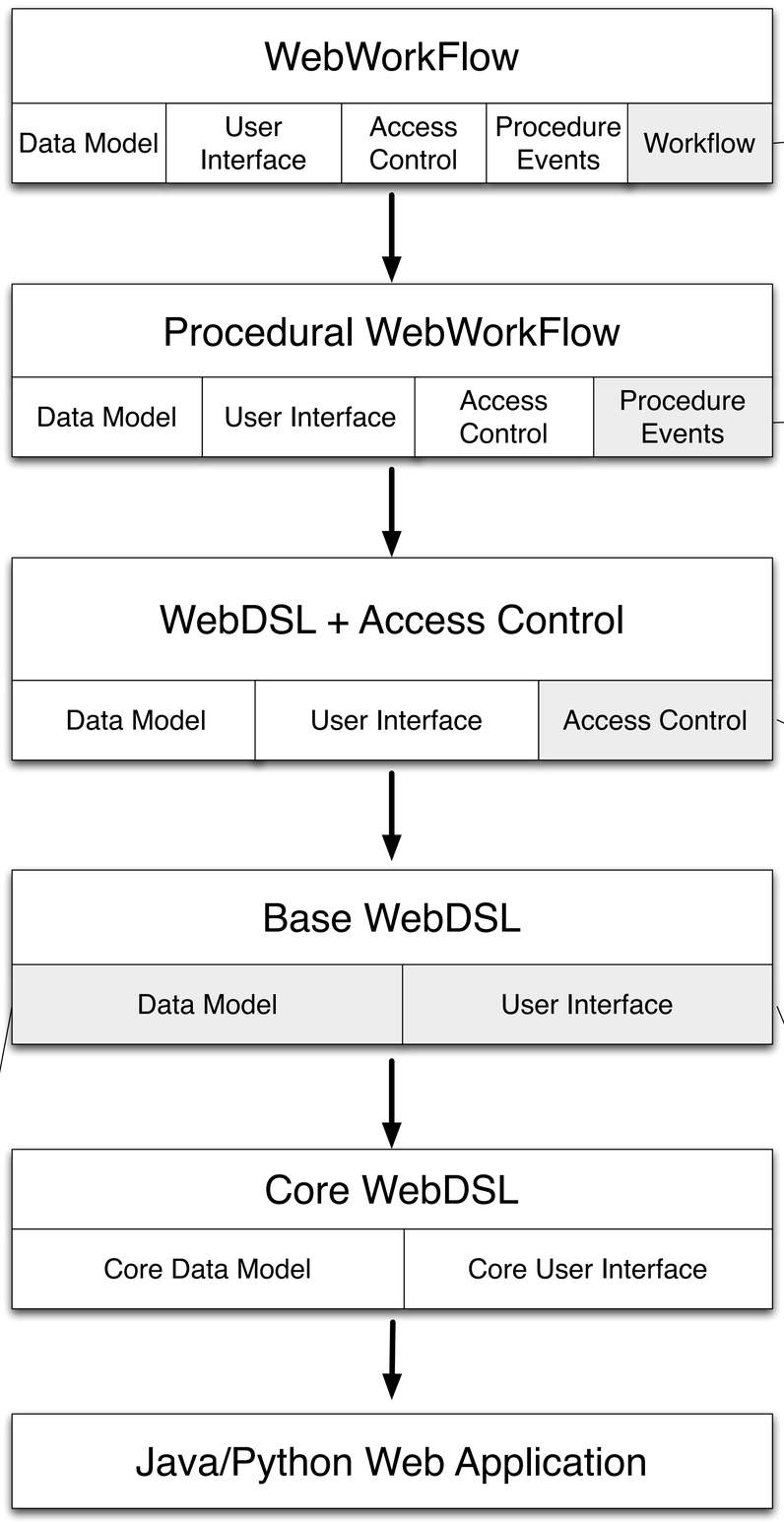
Danny M. Groenewegen, Zef Hemel, Lennart C.L. Kats, Eelco Visser

Problems in web development

- Loose coupling of languages
- No static checking available
- Boilerplate code
- Low expressivity
- Repetition

WebDSL solutions

- Integrated domain-specific sub-languages
- Language concepts for: data models, user interface, access control, workflow
- Generates Java or Python application
- Order of magnitude reduction in application code



Workflow

- High-level business process descriptions
- Specifies actors (who), user interface (view), actions (do)

Generated from workflow

- Task lists
- Status pages
- Navigation

```
procedure meeting(p : ProgressMeeting) {
  process {
    (writeEmployeeView(p) and writeManagerView(p));
    repeat {
      editReport(p);
      (approveReport(p) xor commentReport(p))
    } until finalizeReport(p)
  }
}

procedure commentReport(p : ProgressMeeting) {
  who { principal = p.employee }
  view {
    derive procedurePage from p
      for (view(employee),
           view(report),
           commments)
  }
  do { email(commentNotification(p)); }
}
```

Home | Sign out

Signed in as EmployeeAlice

Comment report

Approve report

Comment report EmployeeAlice

Employee:	EmployeeAlice
Report:	Pretty good.
Comments:	I thought it was great work.

Submit

Access Control

- Rules for: page access, template access, ...
- Separate concern
- Links to inaccessible pages are hidden
- Rules can be combined into policies

```
principal is User with credentials username,
                                   password
access control rules
  rule page editReport(p:ProgressMeeting){
    principal = p.employee.manager
  }
  rule page approveReport(p:ProgressMeeting){
    principal = p.employee
  }
  rule page commentReport(p:ProgressMeeting){
    principal = p.employee
  }
  rule page finalizeReport(p:ProgressMeeting){
    principal = p.employee.manager
  }
```

Signed in as ManagerBob

Edit report

Finalize report

Signed in as EmployeeAlice

Comment report

Approve report

Data Models

- Simple and domain-specific value types (::)
- Composite (<>) and referential (->) associations
- Derived CRUD pages

```
entity User {
  username  :: String (id)
  password  :: Secret
  name      :: String
  manager   -> User
  employees -> Set<User>
  isAdmin   :: Bool
}
entity ProgressMeeting {
  employee     -> User
  employeeView :: Text
  managerView  :: Text
  report       :: Text
  approved     :: Bool
  comment      :: Text
}
```

User: EmployeeAlice

username:	EmployeeAlice
password:	********
name:	EmployeeAlice
manager:	ManagerBob
employees:	
isAdmin:	☐

ProgressMeeting: 1

employee:	EmployeeAlice
employeeView:	Ok, I guess.
managerView:	Great work.
report:	Pretty good.
approved:	☐
comments:	I thought it was great work.

User Interface

- Data presentation
- Markup for structuring pages
- Data entry
- Control of navigation

Comment report EmployeeAlice

Employee:	EmployeeAlice
Report:	Pretty good.
Comments:	I thought it was great work.

Submit

```
define page commentReport(p:ProgressMeeting){
  main()
  define body(){
  header{"Comment report" output(p.employee)}
  form{
    table{
      row{"Employee: " output(p.employee)}
      row{"Report: "   output(p.report)}
      row{"Comments: " input(p.comments)}
    }
    action("Submit",submit())
    action submit(){
      return home();
    }
  }
}
```

Data Manipulation

- Forms with type inferred input elements
- Action definitions with procedural code
- Actions invoked by form button or link

Delft University of Technology

References

E. Visser. *WebDSL: A case study in domain-specific language engineering*. GTTSE (2008)

Z. Hemel, L.C.L. Kats, and E. Visser. *Code Generation by Model Transformation*. ICMT (2008)

E. Visser, *DSLs for the Web (talk)*. OOPSLA (2008)

L.C.L. Kats, M. Bravenboer, and E. Visser. *Mixing Source and Bytecode, a Case for Compilation by Normalization*. OOPSLA (2008)

D. Groenewegen and E. Visser. *Declarative access control for WebDSL: Combining language integration and separation of concerns*. ICWE (2008)

Z. Hemel, R. Verhaaf, and E. Visser. *Webworkflow: An object-oriented workflow modeling language for web applications*. MODELS (2008)

http://www.webdsl.org

This research was supported by NWO/JACQUARD project 638.001.610, *MoDSE: Model-Driven Software Evolution,* and 612.063.512, *TFA: Transformations for Abstractions*